

Rolf Isernhagen
Hartmut Helmke

Softwaretechnik in C und C++ Das Kompendium

Modulare, objektorientierte
und generische Programmierung

ISO-C90, ISO-C99, ISO-C++98,
MS-C++.NET

4., vollständig überarbeitete Auflage

HANSER

Inhaltsverzeichnis

Vorwort zur 4. Auflage	XXI
Vorwort zur 1. Auflage	XXIII
Hinweise für den Leser	XXV
Inhaltsübersicht	XXVII
I C90, C99, Better-C und Modulare Softwareentwicklung	1
1 C90, C99 und Better-C	3
1.1 Einführung	4
1.1.1 Verschiedene Standards und Dialekte	4
1.1.2 Clean-C und Better-C	6
1.1.3 Beispielprogramme	7
1.2 Aufbau der Sprache	13
1.2.1 Lexikalische und syntaktische Struktur	13
1.2.2 Formale Beschreibung (Syntaxnotation)	13
1.2.3 Lexikalische Elemente (Symbole, Token)	14
1.2.4 Trennzeichen (White Spaces) und Kommentare	16
1.2.5 Unterschiede zwischen C99, C90 und C++98	17
1.3 Daten, Operatoren, Ausdrücke, Anweisungen	18
1.3.1 Vordefinierte Datentypen	18
1.3.2 Operatoren für Elementare Datentypen	25
1.3.3 Ausdrücke	30
1.3.4 Prioritäten von Operatoren	31
1.3.5 Explizite und implizite Typkonvertierungen	32
1.3.6 Arbeiten mit Zahlen	33
1.3.7 Arbeiten mit Zeichen	35
1.3.8 Arbeiten mit Booleschen Ausdrücken	36
1.3.9 Aufzählungstypen	37
1.3.10 Ausdrücke als Anweisungen, Mehrfachzuweisungen	38

1.3.11	Bedingte Ausdrücke und bedingte Anweisungen	39
1.3.12	Unterschiede zwischen C99, C90 und C++98	39
1.3.13	Übungen	42
1.4	Steueranweisungen	42
1.4.1	Übersicht	42
1.4.2	Verbundanweisung (compound-statement)	43
1.4.3	Verzweigungen	43
1.4.4	Wiederholungen	45
1.4.5	Marken und Sprünge	48
1.4.6	Zusammenfassung	50
1.4.7	Unterschiede zwischen C99, C90 und C++98	51
1.4.8	Übungen	53
1.5	Funktionen	54
1.5.1	Grundlagen	54
1.5.2	Definition und Deklaration von Funktionen	56
1.5.3	Rekursive Algorithmen und Funktionen	57
1.5.4	<i>Inline</i> -Funktionen	61
1.5.5	Variable Anzahl von Parametern	61
1.5.6	C-Bibliotheksfunktionen	62
1.5.7	Unterschiede zwischen C99, C90 und C++98	63
1.5.8	Übungen	66
1.6	Benutzerdefinierte Datentypen	67
1.6.1	Zeiger, Adressen, Vektoren	67
1.6.2	Strings (Zeichenketten)	73
1.6.3	Mehrdimensionale Vektoren	75
1.6.4	Strukturen	80
1.6.5	Vereinigungen (Unionen)	84
1.6.6	Verwendung des Attributs <code>const</code>	85
1.6.7	Parameter- und Rückgabetypen	86
1.6.8	Zeiger auf Funktionen	87
1.6.9	Zeiger und Zeigerkonvertierungen	92
1.6.10	Unterschiede zwischen C99, C90 und C++98	95
1.6.11	Übungen	102
1.7	Werte- und Zeigersemantik	104
1.7.1	Prinzip der Speicherverwaltung	105
1.7.2	Zeigersemantik beim Arbeiten mit Vektoren	106
1.7.3	Werte- und Zeigersemantik bei Strukturen	107
1.7.4	Werte- und Zeigersemantik bei der Funktionsrückgabe	108
1.7.5	Dynamische Zeichenketten (Strings)	109
1.7.6	Komplexe Deklarationen	111
1.7.7	Unterschiede zwischen C99, C90 und C++98	112
1.7.8	Übungen	117
1.8	Eingabe und Ausgabe von Daten	117
1.8.1	Überblick	117
1.8.2	Arbeiten mit den Standardgeräten	119

1.8.3	Arbeiten mit Textdateien	121
1.8.4	Datenformatierung	124
1.8.5	Dateien mit Blockstruktur	128
1.8.6	Strings als Medium für formatierte Ein- und Ausgabe	131
1.8.7	Anzeige und Abfrage von EA-Zuständen	132
1.8.8	Weitere Funktionen	133
1.8.9	Unterschiede zwischen C99, C90 und C++98	134
1.8.10	Übungen	137
1.9	Präprozessor	138
1.9.1	Einfügen von Dateien	138
1.9.2	Ersetzen von Textstellen	139
1.9.3	Bedingte Einfügung, bedingte Übersetzung	139
1.9.4	Definition von Makros	141
1.9.5	Vordefinierte Namen	143
1.9.6	Sonstiges	143
1.9.7	Zusammenfassung und Bewertung	143
1.9.8	Unterschiede zwischen C99, C90 und C++98	144
1.9.9	Übungen	144
1.10	Programmstruktur und Speicherklassen	145
1.10.1	Dateien, Geltungsbereiche, Lebensdauer	145
1.10.2	Deklarationen und Definitionen	145
1.10.3	Geltungsbereich von Bezeichnern	145
1.10.4	Lebensdauer von Variablenwerten	148
1.10.5	Zusammenfassung und Ergänzungen	150
1.10.6	Module: Definitions- und Implementationsdateien	152
1.10.7	Allgemeiner Aufbau von C-Programmen	156
1.10.8	Unterschiede zwischen C99, C90 und C++98	158
1.10.9	Übungen	161
1.11	Fehler- und Ausnahmebehandlung	162
1.11.1	Verwendung des Makros <code>assert</code>	162
1.11.2	Fehler-Return-Codes bei Funktionen	163
1.11.3	Modulbehaftete globale Statusvariable	163
1.11.4	C++-Exception-Handling-Konzept	164
1.11.5	Übungen	166
2	C-Standardbibliothek	167
2.1	Kurzübersicht über die Gesamtbibliothek	168
2.2	Fehlerdiagnostik: <code><assert.h></code>	168
2.3	Komplexe Arithmetik: <code><complex.h></code>	169
2.4	Zeichenoperationen: <code><ctype.h></code>	170
2.5	Fehlererkennung: <code><errno.h></code>	171
2.6	Gleitkomma-Arithmetik: <code><fenv.h></code>	171
2.7	Grenzwerte für Gleitkommatypen: <code><float.h></code>	171
2.8	Definition von Wortsymbolen: <code><inttypes.h></code>	172
2.9	Definition von Wortsymbolen: <code><iso646.h></code>	173

2.10	Grenzwerte für Ganzzahl-Typen: <code><limits.h></code>	173
2.11	Lokale (nationale) Definitionen: <code><locale.h></code>	174
2.12	Mathematische Funktionen: <code><math.h></code>	174
2.13	Globale Sprünge: <code><setjmp.h></code>	176
2.14	Ausnahmebehandlung: <code><signal.h></code>	177
2.15	Variable Parameterlisten: <code><stdarg.h></code>	177
2.16	Boolesche Arithmetik: <code><stdbool.h></code>	177
2.17	Allgemeine Definitionen: <code><stddef.h></code>	177
2.18	Erweiterte Integer-Typen: <code><stdint.h></code>	178
2.19	Ein- und Ausgabe: <code><stdio.h></code>	179
2.20	Hilfsfunktionen: <code><stdlib.h></code>	182
2.21	String-Operationen: <code><string.h></code>	185
2.22	Typ-generische Makros: <code><tgmath.h></code>	187
2.23	Datum und Zeit: <code><time.h></code>	187
2.24	<i>wide-char</i> und <i>Multibyte</i> -Zeichen: <code><wchar.h></code>	189
2.25	Klassifizierung von <i>wctype</i> -Zeichen: <code><wctype.h></code>	190
3	Modulare Programmierung in C und in Better-C	191
3.1	Module und Modulschnittstellen	192
3.1.1	Softwareentwurf und Modularisierung	192
3.1.2	Geheimnisprinzip, Datenkapselung, Abstrakte Datentypen	193
3.1.3	Realisierung von Abstrakten Datentypen in C	193
3.1.4	Realisierung von generischen Modulen in C	196
3.2	Fallstudie 1: Komplexe Arithmetik	200
3.2.1	Spezifikation	200
3.2.2	Implementierung	201
3.2.3	Anwendung	202
3.3	Fallstudie 2: Strings	203
3.3.1	Spezifikation	203
3.3.2	Implementierung	205
3.3.3	Anwendung	206
3.4	Fallstudie 3: Set (Menge)	207
3.4.1	Allgemeine Spezifikation	207
3.4.2	Modul Set, Version 0: Kleiner Wertebereich	208
3.4.3	Modul Set, Version 1: Vergrößerung des Wertebereichs	209
3.4.4	Modul Set, Version 2: Einstellbarer Wertebereich	212
3.4.5	Modul Set, Version 3: Alternative Realisierung	216
3.4.6	Modul Set, Version 4: Ein generisches Modul	218
3.5	Fallstudie 4: Generische Sortierte Liste	221
3.5.1	Spezifikation	221
3.5.2	Implementierung	223
3.5.3	Anwendung	225
3.5.4	Anpassung an Anwendungen	226
3.6	<i>Better-C</i> : Namensbereiche	235
3.6.1	Grundlagen: Definition und Benutzung	235

3.6.2	Erweiterung von Namensbereichen	236
3.6.3	Alias (Synonym) für Namensbereiche	237
3.6.4	Using-Direktive und Using-Deklaration	238
3.6.5	Anonyme Namensbereiche	239
3.6.6	Anwendung von Namensbereichen	240
3.6.7	Portierung von C-Modulen	242
3.7	Übungen	244
II Algorithmen, Datenstrukturen, ADTs, Container		249
4	Sortieren, Suchen, Rekursion und Zeitkomplexität	251
4.1	Zeitkomplexität	252
4.2	Analyse rekursiver Algorithmen	257
4.3	Einfache Sortieralgorithmen	263
4.4	Schnelle Sortieralgorithmen	266
4.5	Laufzeitmessungen an Sortieralgorithmen	269
4.6	Lineares Suchen	271
4.7	Binäres Suchen	271
4.8	Die Macht der Zeitkomplexität	272
5	Dynamische Datenstrukturen	277
5.1	Lineare Listen, Stapel und Schlangen	278
5.1.1	Zeigernotation und Heapverwaltung	278
5.1.2	Lineare einfach gekettete Listen	280
5.1.3	Lineare doppelt gekettete Listen	283
5.1.4	Stapelspeicher	284
5.1.5	Warteschlange	285
5.1.6	ADT List (Unsortierte Liste)	285
5.1.7	Prioritätswarteschlange	290
5.1.8	Übungen	293
5.1.9	ADT SList (Sortierte Liste)	294
5.2	Binäre Wilde Bäume	298
5.2.1	Grundlagen	298
5.2.2	Traversieren	300
5.2.3	Einfügen	301
5.2.4	Suchen	303
5.2.5	Löschen	303
5.2.6	Übungen	305
5.2.7	ADT Tree (Binärer Suchbaum)	306
5.2.8	Fallstudie <i>Querverweis-Listengenerator</i>	311
5.3	Binäre Ausgeglichene Bäume	316
5.3.1	Definition der Ausgeglichenheit und Konzept	316
5.3.2	Einfügen	318
5.3.3	Löschen	324

5.3.4	ADT Avl (Ausgeglichener Binärer Suchbaum)	328
5.4	Vielweg-Bäume (B-Bäume)	334
5.4.1	Einführung und Konzept	334
5.4.2	Datenstrukturen	337
5.4.3	Traversieren	338
5.4.4	Einfügen	339
5.4.5	Löschen	341
5.4.6	Demoprogramm	344
5.4.7	Übungen	349
5.5	Heap-Strukturen	350
5.6	Hashverfahren	355
5.6.1	Allgemeines	355
5.6.2	Statische Hashverfahren	356
5.6.3	Halbdynamische Hashverfahren	362
5.6.4	Dynamische Hashverfahren	365
5.6.5	Zusammenfassung	368
6	Container-Strukturen	369
6.1	Container-Strukturen, Überblick	370
6.2	Realisierung von Container-Strukturen	370
6.2.1	Stack, Queue und Priorityqueue	371
6.2.2	Set und Multiset	374
6.2.3	Map, Multimap und Vector	375
6.3	Anwendungsbeispiele	380
6.3.1	Test- und Demoprogramm	380
6.3.2	Anwendungsbeispiel <i>Textanalyse</i>	381
6.3.3	Anwendungsbeispiel <i>Anagramme</i>	384
6.4	Übungen	388
7	Automaten in der Softwaretechnik (ausgelagert auf die CD-ROM)	963
III	Die Sprache C++ und ihre Softwaretechniken	391
8	Klassen als Abstrakte Datentypen	393
8.1	Von C nach C++	394
8.2	Einführung, Klassen als Abstrakte Datentypen	401
8.2.1	Von der Struktur zur Klasse	401
8.2.2	Grundlagen	403
8.3	Konstruktoren und Destruktoren	408
8.3.1	Verschiedene Arten von Konstruktoren	408
8.3.2	Implizite Erzeugung von Konstruktoren	410
8.3.3	Die Bedeutung des Kopierkonstruktors	410
8.3.4	Die Bedeutung von Umwandlungskonstruktoren	411
8.3.5	Destruktoren, implizite Erzeugung von Destruktoren	412
8.4	Der Zuweisungsoperator	412

8.4.1	Implizite Erzeugung	412
8.4.2	Implementierung – das Prinzip	413
8.5	Zugriffsrechte	414
8.6	Minimale Standardschnittstelle und <i>Nice Class</i>	414
8.7	Konstante Methoden konstante Objekte	417
8.8	UML-Klassendiagramme: <i>Modelle der Programmstruktur</i>	419
8.8.1	Modellierung von Klassen mit UML	420
8.8.2	Modellierung von Beziehungen zwischen Klassen	421
8.9	<i>Mitglieder</i> und <i>Freunde</i>	424
8.9.1	<i>Friend</i> -Beziehungen	424
8.9.2	Beispiel: <i>Friend</i> -Klassen	425
8.9.3	Beispiel: Komplexe Arithmetik, Version 0	425
8.10	Überladen von Operatoren	427
8.10.1	Beispiel: Komplexe Arithmetik, Version 1	427
8.10.2	Hinweise zum Überladen von Operatoren	431
8.10.3	Überladbare und nicht überladbare Operatoren	431
8.10.4	Überladen von Konvertierungsoperatoren	432
8.11	Dynamische verwaltete Daten in Klassen	433
8.11.1	Beispiel: Komplexe Arithmetik, Version 2	433
8.11.2	Implizite Verwendung des Kopierkonstruktors	434
8.11.3	Überladen des Zuweisungsoperators	435
8.12	Effizienzbetrachtungen: Laufzeiteffizienz	436
8.12.1	Grundlegende Überlegungen	436
8.12.2	Beispiel: Komplexe Arithmetik, Version 3	437
8.13	Statische Elemente in einer Klasse	438
8.14	Schachtelung von Klassen	441
8.15	Funktionsobjekte	443
8.16	Zusammenfassung: Hinweise und Regeln	446
8.17	Übungen	447
9	Objektorientierte Programmierung	453
9.1	Vererbung: <i>Erweiterung von Klassen</i>	454
9.1.1	Grundlagen	454
9.1.2	Konstruktoren, Destruktor und Zuweisungsoperator	458
9.1.3	Typkompatibilität in Klassenhierarchien	462
9.1.4	Virtuelle Funktionen	463
9.1.5	Öffentliches und Privates Erben	466
9.1.6	Privates Erben und Benutzen	467
9.1.7	<i>Rein</i> Virtuelle Funktionen und Abstrakte Klassen	469
9.1.8	UML-Klassendiagramme: <i>Vererbungshierarchien</i>	472
9.1.9	Übungen	474
9.2	Polymorphie: <i>Dynamisches Binden</i>	476
9.2.1	Dynamischer Typ, dynamisches Binden	476
9.2.2	Heterogene Datenstrukturen	480
9.2.3	Virtuelle Destruktoren	481

9.2.4	Ein Klassensystem für grafische Objekte, Vers. 1	483
9.2.5	Objektschachtelung: Grafische Objekte, Vers. 2	485
9.2.6	Zwei Aspekte der Polymorphie: <i>heterogen</i> und <i>generisch</i>	488
9.2.7	Polymorphe Container-Strukturen	492
9.2.8	Übungen	506
9.3	Die richtige Anwendung der Vererbung	508
9.3.1	Verschiedene Sichten und Probleme	508
9.3.2	Schnittstellen-Klassen	510
9.3.3	Abstrakte und nicht abstrakte Basisklassen	512
9.3.4	Zuweisungsoperatoren in Klassen mit polymorphen Zeigern	516
9.3.5	Übungen	518
9.4	Mehrfachvererbung	519
9.4.1	Grundlagen	519
9.4.2	Verwendung nicht virtueller Basisklassen	520
9.4.3	Verwendung virtueller Basisklassen	522
9.4.4	Virtuelle Methoden in virtuellen Basisklassen	527
9.4.5	Fallstudie <i>Querverweis-Listengenerator</i>	529
9.4.6	Übungen	534
9.5	Typkonvertierung, Laufzeit-Typinformation	535
9.5.1	Die Grundidee	535
9.5.2	Syntax und Semantik der Sprachkonstrukte	537
9.5.3	Zusammenfassung	539
9.6	Ausnahmebehandlung (Exception-Handling)	540
9.6.1	Grundlagen	541
9.6.2	Syntax und Semantik der Sprachkonstrukte	555
9.6.3	Zusammenfassung und Empfehlungen	557
9.6.4	Übungen	559
9.7	Besondere Programmiertechniken	560
9.7.1	Referenzzählung (Reference-Counting)	560
9.7.2	Up-Calls	568
9.7.3	Botschaften und Botschaftsinterpretier	570
10	Generische Programmierung	575
10.1	Klassen-Templates: <i>Generische Strukturen</i>	576
10.1.1	Grundlagen	576
10.1.2	UML-Klassendiagramme: <i>Klassenschablonen</i>	583
10.1.3	Eine generische Container-Bibliothek	584
10.1.4	Spezialisierung von Templates	590
10.1.5	Eine spezialisierte Container-Bibliothek	593
10.1.6	Exception Handling am Beispiel eines <i>Stack-Templates</i>	596
10.1.7	Übungen	601
10.2	Funktions- und Element-Templates	602
10.2.1	Funktions-Templates	602
10.2.2	Element-Templates	603
10.3	Besondere Programmiertechniken	604

10.3.1	Smart Pointer	604
10.3.2	Iteratoren	609
10.3.3	Generische Algorithmen	617
10.4	Entwicklung einer kleinen Container-Bibliothek	621
10.4.1	Konzept und Überblick	621
10.4.2	Die Klassen List, SList und Tree	621
10.4.3	Die Containerklassen Stack, Queue und Prio	624
10.4.4	Set und Multiset	626
10.4.5	Die Containerklassen Map, Multimap und Vector	627
10.4.6	Test- und Demoprogramm für die Containerklassen	631
10.5	Template-Metaprogrammierung	632
10.5.1	Compile-Zeit-Berechnungen	633
10.5.2	Laufzeit- und Compile-Zeit-Polymorphismus	636
11	C++-Standardbibliothek	639
11.1	Übersicht	640
11.2	Allgemeine Dienste	642
11.2.1	Vergleichsoperatoren	643
11.2.2	Paare	643
11.2.3	Auto-Zeiger	644
11.2.4	Übungen	645
11.3	Strings	646
11.3.1	Konzept und Übersicht	646
11.3.2	Funktionalität	651
11.4	Ein- und Ausgabe (EA)	662
11.4.1	Vorteile des neuen EA-Konzeptes	662
11.4.2	EA im Standardformat mit Standardgeräten	663
11.4.3	Struktur & Funktionalität der Stream-IO-Bibliothek	665
11.4.4	Formatierung	671
11.4.5	EA-Funktionen	677
11.4.6	Arbeiten mit Dateien	680
11.4.7	Strings als Ein- und Ausgabemedium	687
11.4.8	Anzeige & Abfrage von Zuständen, Fehlerbehandlung	689
11.4.9	Eigene Definitionen, weitere Funktionen	693
11.5	Container	703
11.5.1	Das Konzept der <i>Standard Template Library (STL)</i>	703
11.5.2	Einführung	704
11.5.3	Gemeinsame Definitionen	709
11.5.4	Container für Sequenzen	713
11.5.5	Containeradapter für Sequenzen	721
11.5.6	Container für Assoziationen	725
11.5.7	Die Containerklasse <code>bitset</code>	733
11.5.8	Übungen	737
11.6	Iteratoren	738
11.6.1	Operationen und Klassifizierung	738

11.6.2	Iteratoren als Container-Schnittstellen	739
11.6.3	Reverse-Iteratoren	743
11.6.4	Insert-Iteratoren	745
11.6.5	Stream-Iteratoren	747
11.6.6	Selbst definierte Iteratoren (Iterator-Adapter)	750
11.7	Funktionsobjekte	755
11.7.1	Verwendung von Funktionsobjekten	755
11.7.2	Funktionsobjekte der Standardbibliothek	757
11.7.3	Funktionsadapter der Standardbibliothek	760
11.8	Algorithmen: globale generische Funktionen	763
11.8.1	Nicht modifizierende sequenzielle Operationen	765
11.8.2	Modifizierende sequenzielle Operationen	769
11.8.3	Sortieren, Suchen und verwandte Operationen	775
11.8.4	Mengenoperationen	780
11.8.5	Heap-bezogene Operationen	783
11.8.6	Minimax bestimmen und lexikografische Vergleiche	786
11.8.7	Permutieren	789
11.8.8	Anwendungsbeispiel <i>Textanalyse</i>	791
11.8.9	Übungen	794
11.9	Weitere Dienstleistungen	796
11.9.1	Ausnahmebehandlung	796
11.9.2	Grenzwerte der Implementierung	799
11.9.3	Speichermodelle	800
11.9.4	Internationalisierung	801
11.9.5	Numerik	802
IV	Die Sprache C++.NET und ihre Bibliothek	805
12	C++.NET: <i>Verwalteter Code</i>, Objektmodell und Bibliothek	807
12.1	Von C++98 nach C++.NET – ein Überblick	808
12.1.1	Das <i>Common Language Runtime</i> -System	808
12.1.2	Managed Code – Arbeiten mit <i>verwalteten Objekten</i>	811
12.1.3	.NET-Datentypen und .NET Framework Library	818
12.2	Klassen im Namensraum System	820
12.2.1	Die Klasse Object und das .NET-Programmiermodell	820
12.2.2	Die Klasse Console – Arbeiten mit der Konsole-E/A	823
12.2.3	Arbeiten mit Dateien	825
12.2.4	Die Klasse String	826
12.2.5	Container-Klassen in System::Collections	831
12.2.6	Übungen	834
12.3	Verwaltete Objekte unter .NET	834
12.3.1	Das Schlüsselwort __gc	834
12.3.2	Das Schlüsselwort __nogc	835
12.3.3	Das Schlüsselwort __value	837

12.3.4	Das Schlüsselwort <code>__box</code> , Boxing und Unboxing	838
12.3.5	Das Schlüsselwort <code>__pin</code>	840
12.4	Vererbung und Typkontrolle unter .NET	842
12.4.1	Das Schlüsselwort <code>__abstract</code>	842
12.4.2	Das Schlüsselwort <code>__interface</code>	844
12.4.3	Das Schlüsselwort <code>__sealed</code>	848
12.4.4	Das Schlüsselwort <code>__typeof</code>	848
12.4.5	Das Schlüsselwort <code>__try_cast</code>	849
12.5	Ereignisbehandlung unter .NET	850
12.5.1	Das Schlüsselwort <code>__delegate</code>	851
12.5.2	Das Schlüsselwort <code>__event</code>	852
12.5.3	Anwendung für <i>Delegates</i>	854
12.5.4	Anwendung für Events	858
12.6	Weitere .NET-Schlüsselwörter	862
12.6.1	Das Schlüsselwort <code>__property</code>	862
12.6.2	Das Schlüsselwort <code>__identifizier</code>	865
12.7	.NET-Datentypen	866
12.7.1	Einfache .NET-Datentypen	866
12.7.2	Verwaltete Arrays	868
12.8	Operatoren überladen unter .NET	871
12.9	Übungen	873
13	Programmieren mit <i>Managed Extensions</i>	875
13.1	Fallstudie: Ein Botschaftsinterpret	876
13.2	Mischen von <i>verwaltetem</i> und <i>nicht verwaltetem</i> Code	878
13.2.1	Speicherverwaltung	880
13.2.2	Destruktor und <i>verwaltete Objekte</i>	884
13.2.3	Beispiel	885
13.2.4	<i>Verwaltete Zeiger</i> in <i>nicht verwalteten Typen</i>	890
13.3	Ausnahmebehandlung	894
13.4	Die STL in Zusammenhang mit <i>Managed Code</i>	901
13.5	Details der C++.NET-Garbage Collection	909
13.6	Portabler Code trotz Managed Extensions?	916
13.6.1	Grundsatz-Überlegungen	916
13.6.2	Portabilität der neuen Schlüsselwörter	917
13.6.3	Objekte des Namensraums <code>System</code>	921
13.6.4	Weitere Aspekte der Portabilität	922
13.7	Übungen	928
	Literaturverzeichnis	929
	Stichwortverzeichnis	933



Die folgenden Teile befinden sich auf der CD-ROM



Texte auf der CD-ROM (<i>Buch-CD.ps</i> und <i>Buch-CD.pdf</i>)	947
C/C++-Textverzeichnis	949
Abbildungsverzeichnis	957
Tabellenverzeichnis	961
7 Automaten in der Softwaretechnik	963
7.1 Automaten und Sprachen	964
7.2 Endliche Automaten und Reguläre Sprachen	964
7.3 Lexikalische Analyse, Scanner	967
7.3.1 Scanner für Zahlen	969
7.3.2 Scanner für Kommentare	970
7.3.3 Ein Scanner für die kleine Sprache ML2	971
7.4 Kellerautomaten und Kontextfreie Sprachen	975
7.5 Syntaktische Analyse, Parser	979
7.5.1 EBNF-Notation und Syntaxdiagramme	979
7.5.2 Parser	981
7.5.3 Die Sprachen ML1 und ML2 und ihre Parser	986
Anhänge: Lösungen zu den Übungen	1
A Lösungen zu Kapitel 1	3
A.1 Lösungen zu Abschnitt 1.3	4
A.2 Lösungen zu Abschnitt 1.4	4
A.3 Lösungen zu Abschnitt 1.5	6
A.4 Lösungen zu Abschnitt 1.6	17
A.5 Lösungen zu Abschnitt 1.7	22
A.6 Lösungen zu Abschnitt 1.8	26
A.7 Lösungen zu Abschnitt 1.9	40
A.8 Lösungen zu Abschnitt 1.10	43
A.9 Lösungen zu Abschnitt 1.11	52
B Lösungen zu Kapitel 3	57
C Lösungen zu Kapitel 5	77
C.1 Lösungen zu Abschnitt 5.1	78
C.2 Lösungen zu Abschnitt 5.2	81
C.3 Lösungen zu Abschnitt 5.4	88
D Lösungen zu Kapitel 6	97
E Lösungen zu Kapitel 8	105

F	Lösungen zu Kapitel 9	141
F.1	Lösungen zu Abschnitt 9.1	142
F.2	Lösungen zu Abschnitt 9.2	149
F.3	Lösungen zu Abschnitt 9.3	152
F.4	Lösungen zu Abschnitt 9.4	158
F.5	Lösungen zu Abschnitt 9.6	158
G	Lösungen zu Kapitel 10	163
H	Lösungen zu Kapitel 11	167
H.1	Lösungen zu Abschnitt 11.2	168
H.2	Lösungen zu Abschnitt 11.5	178
H.3	Lösungen zu Abschnitt 11.8	191
I	Lösungen zu Kapitel 12	223
J	Lösungen zu Kapitel 13	233